Tiger Graphics

CEvoS

CEMoS **E**volutionary **S**ystem

Version **4.1**

User's Manual

C. Kohlmeier & F. Hamberg

September 13, 2012

Contents

1 Introduction **5**

1.1 The idea of parameter adaption 6
1.2 Requirements and Installation of CEvoS 6
1.3 Defining a parameter set for the optimization – the file cevos.cev 6
1.4 Starting CEvoS 9
1.4.1 Starting CEvoS in batch mode 10
1.5 The graphical user interface (GUI) 10
1.5.1 Info window 1 10
1.5.2 Info window 2 10
1.5.3 Best distance and Runs 10
1.5.4 Parameters 10
1.5.5 [Reference File] 11
1.5.6 [Start !] 11
1.5.7 [Calculate Distances] 11
1.5.8 Distances 11
1.5.9 Pointwise distances 12
1.5.10 L2 distances 13
1.5.11 Positive Par 14
1.5.12 Change all active 14
1.5.13 Optimization 15
1.5.14 [Stop Rnd] 18
1.5.15 [Reset to Best] 18
1.5.16 [Reset to AbsBest] 18
1.5.17 [Rename] 18
1.5.18 [Reset] 18
1.5.19 Tiger Graphics MoViE 18
1.5.20 [Exit] 18
1.5.21 [CEV File] 19
1.5.22 [CIN File] 19
1.5.23 [Reference File] 19
1.5.24 [Stop Running Model] 19
1.6 The file cevos.init 20

2 Output Files **23**
2.1 best.outc, best.info, best.log 23
2.2 best.dist 23
2.3 best.bat and best.cin 23
2.4 The files absbest* 23

3 Converting field data to comparable files **25**

4 Auxillary scripts **27**
4.1 def2cev 27
4.2 cev2ser 28

License

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see ¡http://www.gnu.org/licenses/¿.

1 Introduction

CEvoS – **CEMoS** Evolution System– is an extension of **CEMoS** (Hamberg & Kohlmeier, 2004) which allows an optimization of model parameter values in respect to a given reference run. This reference run can either be another model run or a preprocessed run of field data. The package **MoViE** (Kohlmeier & Hamberg, 2004) provides a preprocessing routine for transforming field data to the structure of a model run
(prepare tool → [**data** → **outc**]).

The basic idea is to determine the distance of a reference run and a simulation run depending on the actual parameter set and to minimize this distance by varying the parameters. These distance is often also called cost function. The base for the calculation of the distance are all variables which are stored in both, the reference run and the simulation run. The implemented calculation of the distance is described in section 1.5.8. **CEMoS** allows the variation of all parameters defined in the file `cevos.dat` which must exist defined in a directory `<model_dir>/cevos`. This parameters are either varied by a defined series or by random. After every calculation the resulting distance is compared to the former distance. The best parameterization is stored in `bestpar.bat`, the best distance in `bestdist.dat` and the run itself in `best.outc`. For the decision which is the best distance three optimization algorithms are implemented(section 1.5.13).

If the system has been started by `Start Rnd` (section 1.5.13) one parameter defined in `cevos.dat` is varied randomly by 10% of the value at maximum. The optimization cycle will run until `Stop Rnd` is pressed. If the system is restarted again the optimization process will start with the last best parameterization.

CEvoS also allows sensitivity analyses. For this a simulation series can be started where a selected parameter is varied by a pre detrmined increment in apre determined range. If a series has been calculated **CEvoS** stops automatically after this series.The distances to the reference run are stored.

All simulation runs are stored in `<model_dir>/cevos`. The last run is always named `result0.outc`. If it is the best run it is copied to `best.outc`. In case of a random start only the last run is kept to avoid storage overflow.

WARNING: Other files than files used by CEvoS in the directory `cevos` are deleted during the run of CEvoS. Do not store any information in the directory `cevos`!!

The simulation of a model is similar to the simulation within the **CEMTK** shell. All parameter changes are read from the `cemos.par` resp `def`-files. Alternatively can the simulation be started using a `cin`-file. In both cases the values of the parameters defined in `cevos.cev` are overruled by the values defined in `cevos.cev`.

1.1 The idea of parameter adaption

The optimization of parameter can be summarited as follows.

1. Defining the parameters to be optimized (see 1.3)
2. Defining a target; a reference run. This reference run can either be another model run or a preprocessed run of field data (see 3)
3. Determing a cost function or distance function (see 1.5.8)
4. Determing an optimization method and escape criteria (see 1.5.13)

1.2 Requirements and Installation of CEvoS

CEvoS is part of **TigerGraphics TigerPack** and an addendum of **CEMoS** and **MoViE**. For installtion see **CEMoS** manual. After the installation of **CEMoS/CEMTK** and **MoViE**, **CEvoS** is available and ready for use. Before starting, the parameters to be varied have to be defined.

1.3 Defining a parameter set for the optimization – the file cevos.cev

Before starting **CEvoS** the directory `<model_dir>/cevos` must exist containin at least on file with the extension `cev`. This file specifies the model parameters which should be trained. If the file is named `cevos.cev` it is read by default. Files with other names can be selected by [CEV Files]. There are to alternatives for the syntax of this file:

1. `def`-file syntax Parameters are set similar to the settings in the def files, f.e.

```
a=3.0; /* Parameter a */
b=2.5; /* Parameter b */
```

The comments are evaluated as description. In this notation only the column `value` in the **CEvoS** window is valid. No series run is possible. In the case of an optimization run, no interval is evaluated. In the case of indexed variables the values must be given separately, f.e.:

```
a[1]=1.5; /* Parameter a[1] */
a[2]=3.5; /* Parameter a[2] */
```

2. `%series` syntax A range for the parameters can be given similar to the series in bat files., f.e.

   ```
   a:0.1:0.5:0.05;   /* Parameter a */
   ```

   where the first identifier is the parameter name, the second the minimum value, the third the maximum value and the fourth the increment. In this notation the column `from` contains the minimum value, `to` the maximum value and `runs` the number of runs derived from the increment.The actual value is set to the minimum value. With these settings case series runs and optimization runs are possible.

   If an optimization run is started the parameters are varied within the interval defined by the values in the columns `from` and `to`. If `from` and `to` are equal no limitation for the parameter variation is evaluated (exception :`Positive Par` is enabled 1.5.11).

   If the optimization run should be started with a special initial value it can be defined in the sixth column

   ```
   a:0.1:0.5:0.05:/* Parameter a */: 0.12;
   ```

   In this case the simulation starts with this predefined starting value, otherwise it starts with the avearge of `from` and `to`. If the randomly chosen parameter exceeds the interval, the parameter is set to the predefined starting value (if existing) or to the average of `from` and `to`.

   In the case of the optimization run the parameter for the increment is not evaluated (but must be set for consistency).

Two auxillary scrips exist to get cev files from def file resp. to define a series file with ranges (see4).

1.4 Starting CEvoS

To start **CEvoS**, go to the directory where an existing and compiled **CEMoS** model (the file `cemos.par`) is located and give the command

`cev cemos.par`

The **CEvoS** windows appear (figure 1.1).

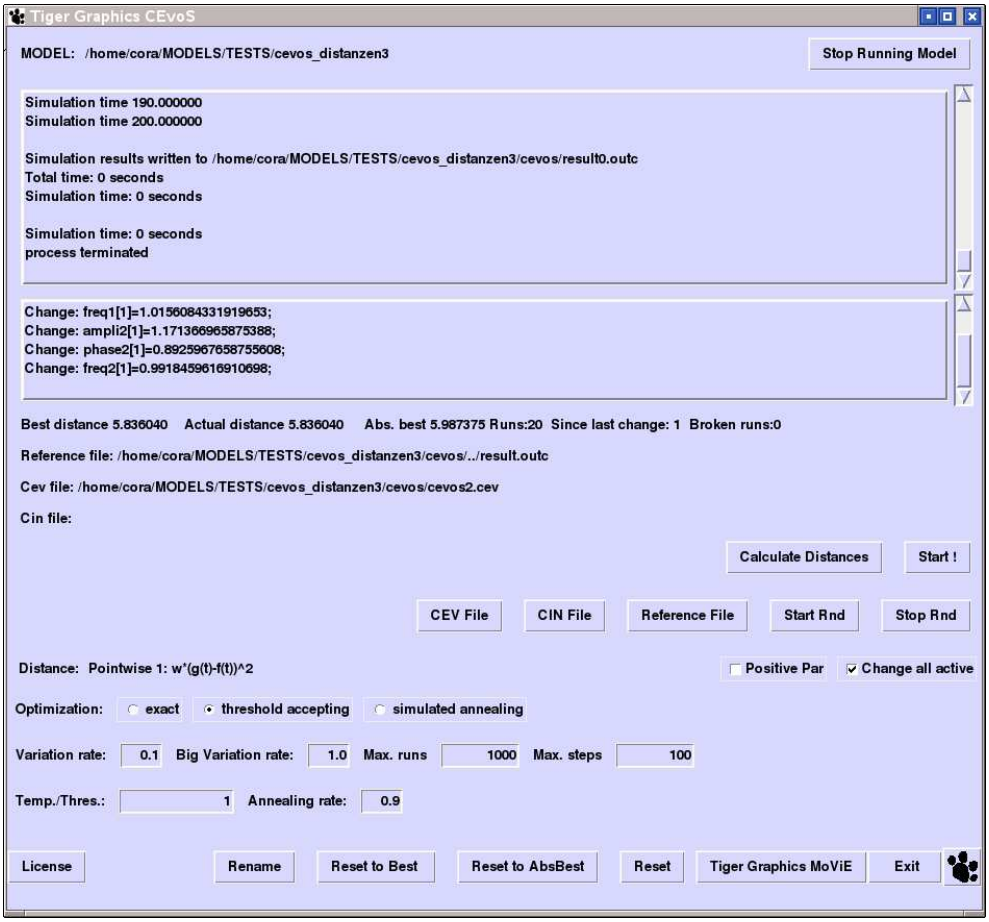

Figure 1.1: Main window of **CEvoS**.

A single simulation with the parameterization shown in the `value` column can be started with `[Start!]`. If `series` is enabled a a parameter can be selected for a series run. The start value, the end value and the number of runs must be set . `[Start!]` starts the series if `Enable Series` is enabled. All series runs will be simulated automatically. The best simulation will be stored as `best.outc`. `[Reset to Best]` will set the parameters to the last best parameter set.

An optimization run can be started by `Start Rnd` The parameter values are modified by random (in the given range) (see 1.5.13).

1.4.1 Starting CEvoS in batch mode

It is also possible to start **CEvoS** without GUI.To start **CEvoS** in batch mode without GUI, go to the directory where an existing and compiled **CEMoS** model (the file `cemos.par`) is located and give the command

`cev`

In this case the evolution run is started. To interrupt the optimization, press `<Ctrl>-c`. The optimization run can be customized by the settings in the file `cevos.init` (see 1.6).

1.5 The graphical user interface (GUI)

1.5.1 Info window 1

Within this window all information during simulation etc. is shown.

1.5.2 Info window 2

Within this window all parameter changes of the actual simulation are shown.

1.5.3 Best distance and Runs

The actual best distance, the actual absolute best distance is displayed. The number of runs, the number of runs after the last change of the best distance and the total number of broken runs (runs which coud not be finished for any reason) is displayed. The best distance, the difftype and the number of runs is stored in the file `best.dist` (see 2.2).

1.5.4 Parameters

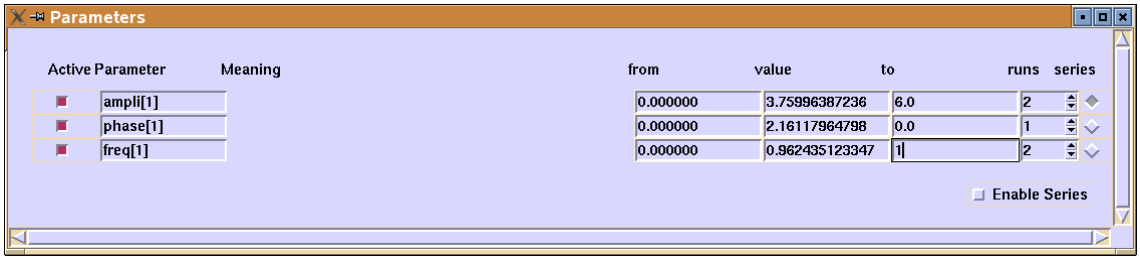

Figure 1.2: Parameter window of **CEvoS**.

The parameters are shown in the parameter window (figure 1.2. Depending on the settings in the file `cevos.dat` each line represents one parameter. The

first column contains a checkbutton. The parmeter can be enabled or disabled. If the parmeter is disabled no variation of its value takes place. The second column contains the name of the parameter , the third its meaning as defined in the comment, the column `from` contains the minimum value of the parameters. The column `value` contains the actual value of the parameter. The column `to` contains the maximum value. `runs` and `series` are only valid if `Enable Series` is enabled.

1.5.5 [Reference File]

A `reference`-file must be determined for the calculation of the distances. This reference file must have the structure of a **CEMoS** result file (see 3). The distances are calculated for all variables which are stored in the reference file and in the simulation result files. A reference file can also be made from field data. For further information see 3.

1.5.6 [Start !]

Starts either a single simulation with the actual parameter values or a series (if enabled).

1.5.7 [Calculate Distances]

After a simulation the difference between the reference file (if determined) and the actual run resp. runs in case of a series are printed in the info window. Additionally, the differences for each variable are stored in files named
`<reference_name>_<name>_m<j>.dist`,
where `<reference_name>` is the name of the selected reference fileand `<name>` is the name of the result file and `<j>` the index of the distance.

1.5.8 Distances

Two types of distances are available: pointwise distances and L2 distances. The pointwise distances compare the reference file and the actual simulation at every stored time step. The L2 distances make Fourier analyses of both, the reference file and the actual run and compare the spectral values. It is either a pointwise or a L2 distance active. The active distances is displayed.
The distances are not normed to a common maximum value so that the values calculated with different distance types are not comparable.Thus, if a previous

distance calculation is done with a different distance type, a message is given and the simulation stops. In this case the stored best distance and best files must be deleted before restating the simulation by [Reset].

1.5.9 Pointwise distances

The pointwise distance of the reference file and the simulation run is determined in two steps. First, the distance between every single stored variable is determined for the stored time window and normed by the number of time steps. Afterwards the resulting distance values for the different variables are averaged. The resulting number is the distance of the two files. The intermediate distances for the variables are stored in the file by `[Calculate Distances]`. Actually six pointwise distance types are available. Assuming that x_{ki}, $k = 1..K, i = 1..N$ are the results from a reference file x for a variable k and y_{ki} the results from the simulation y, than the distances $d(x, y)$ are calculated as follows:

1. $$d_1(x,y) = \frac{1}{2}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{N}\sum_{i=1}^{N} w_k(x_{ki}-y_{ki})^2 \quad w_k = \frac{\max_k \overline{x_k}}{\overline{x_k}}$$

   where $\overline{x_k}$ is the time average of variable k of the reference file (Spitz *et al.* , 1998).

2. $$d_2(x,y) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N}\sum_{i=1}^{N}\frac{|x_{ki}-y_{ki}|}{|x_{ki}+y_{ki}|}$$

3. $$d_3(x,y) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N}\sum_{i=1}^{N}\frac{|x_{ki}-y_{ki}|}{\min(|x_{ki}|,|y_{ki}|)}$$

4. $$d_4(x,y) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N}\sum_{i=1}^{N}\frac{|x_{ki}-y_{ki}|}{|x_{ki}|+|y_{ki}|-|x_{ki}-y_{ki}|}$$

5. $$d_5(x,y) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N}\sum_{i=1}^{N}\ln|\frac{x_{ki}}{y_{ki}}|$$

6. $$d_6(x,y) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N}\sum_{i=1}^{N}(\max(|\frac{x_{ki}}{y_{ki}}|,|\frac{y_{ki}}{x_{ki}}|)-1)$$

The distances are not normed to a common maximum value so that the values calculated with different distance types are not comparable.Thus, if a previous distance calculation is done with a different distance type, a message is given and the simulation stops. In this case the stored best distance and best files must be deleted before restating the simulation by `[Reset]`.

The number of the distance is the same as in the distance file `bestdist.dat`.

1.5.10 L2 distances

The so called L2 distances work on the spectral values of the runs. The transformation of the signal into spectral values allows the comparison of to results in a manner which exceeds the pointwise method by far. The comparison of simulation results and field measurements is a complicated task. The normal way is to plot both within one graph and to decide with human intuition the degree of correspondence. This is of course a very subjective decision because from psychology it is well known that people tend to see what they expect or like to see.

Such it is necessary to give an assistance to the human intuition for the exploitation of the characteristic differences and correspondings of the theoretical results and the "real life". This is only possible if not only single points are compared but the time series are compared at a whole. Only then f.e. phase shifts can be taken into account.

The L2 distances are calculated as follows: In a first step a fast Fourier analyses of the reference file and the simulation run is made resulting in `_pk.outc` file containing the spectral values for every stored variable. This method works only successfully if the data are periodic in time and the stored time interval covers the period.

The maximum number of complex Fourier coefficients k is – concerning the sampling theorem (Stearns (1984))– half the number of samples. Assuming N samples, the Fourier coefficients are given by a vector $g := (g_0, g_1, ..., g_{2k}) \in \mathbb{R}^{2k+1}$ with $k = \left[\frac{N}{2}\right] - 1$. Thus the original value of a variable x at time t is given by

$$x(t) = \frac{1}{\sqrt{2\pi}} g_0 + \frac{1}{\sqrt{\pi}} \sum_{l=1}^{k} \left(g_{2l-1} \cos(lt) + g_{2l} \sin(lt)\right)$$

The norm of a time series x with the Fourier coefficients $f_0, f_1, ..., f_{2k}$ (which is an element of the room L2 – hence the name L2 distances) is given by

$$\|x\| = \sqrt{\sum_{l=0}^{2k} f_l^2}$$

and the scalar product of two time series x and y with the Fourier coefficients $f_0, f_1, ..., f_{2k}$ resp. $g_0, f_1, ..., g_{2k}$ is given by

$$< x, y >= f_0 g_0 + \sum_{l=0}^{2k} f_l g_l \cos(\gamma_l - \varphi_l)$$

with $\gamma_l = \arctan\left(\frac{f_{2l}}{f_{2l-1}}\right)$ and $\varphi_l = \arctan\left(\frac{g_{2l}}{g_{2l-1}}\right)$.
The following distance functions are implemented:

1. $d_1(f,g) = ||f-g||/||f+g||$
2. $d_2(f,g) = ||f-g||/min(||f||,||g||)$
3. $d_3(f,g) = ||f-g||/(||g|| + ||f|| - ||f-g||)$
4. $d_4(f,g) = 1 - < f,g > /(||f||\ ||g||)$
5. $d_5(f,g) = (||f||\ ||g|| - < f,g >)/(||f||\ ||g|| + < f,g >)$
6. $d_6(f,g) = \arccos(< f,g >)/(||f||\ ||g||))$
7. $d_7(f,g) = |\ln(||f||/||g||)|$
8. $d_8(f,g) = \max(||f||/||g||, ||g||/||f||) - 1$
9. $d_9(f,g) = \lambda_1 d_6(f,g) + \lambda_2 d_7(f,g)\ ,\ \lambda_1 = \lambda_2 = 1$

The distances $d_1 - d_3$, d_7 and d_8 focus on differences in the amplitudes, while distances $d_4 - d_6$ measure the phase shift. All distances are invariant concerning rotation and stretching, some are definite, surjective or fulfill the triangle inequality. Only d_9 fulfills all these properties. Krug (1992) has shown that d_9 is nearest to the human estimation.
WARNING: The preparation for the calculation of L2 distances (matrix inversion) can take a very long time after the first model run !! The inverted matrix is stored so that the following calculations are quite faster.

1.5.11 Positive Par

If this option is set the parameter variations allows only non negative values.

1.5.12 Change all active

If this option is set the parameter values of all active parameters are changed by random. Otherwise only one parameter value is changed by random. For every run the parameter to be changed is selected by random

1.5.13 Optimization

The decision if the actual run is better than the actual best run can be done in different ways. The obvious method is to compare their distances (costs). relative to the reference file:

Exact method (EX)

The distance D_{act} is compared to the distance of the actual best run with distance D_{best} The criterion for accepting the actual run as new best run is given by

$$D_{act} < D_{best}$$

To avoid that the optimization stucks in a local minimum the number of runs is bounded in two ways:
If the number of runs without finding a new best run exceeds `Max. steps`, the variation of parameters is increased (see 1.5.13).
If the number of runs without finding a new best run exceeds `Max. runs`, the system is new initialized by the values in the `*.cev file`.

Threshold accepting (TA)

The exact method has the disadvantage that local minima of the cost function cannot be left. This can be reached if a run with a slighty worse distance is accepted:

$$D_{act} < D_{best} \cdot (1 + T)$$

where the threshold T is a positive number less than 1. Thus, the actual run is accepted as new best run although its costs are higher than the cost of the actual best run.
To get convergence, the threshold value `T` will be gradually decreased after a pre-defined number of runs (see 1.5.13).

Simulated annealing (SA)

The simulated annealing optimization Morgan (2001) has its roots in metallurgy. Heat causes the atoms in a metal to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly

through states of higher energy. Slow cooling gives them more chances of finding configurations with lower internal energy than the initial one.
By analogy with this physical process, each step of the SA algorithm replaces the actual best run by the actual run with a probability that depends on a global parameter T (called the temperature). T is gradually decreased during the process (annealing). The best run changes almost randomly when T is large, but increasingly "downhill" as T goes to zero. The acceptance of a run which is not absolutely the actual best one takes care that local minima can be left.
The criterion (Metropolis criterion) for accepting the actual run as new best run is given by

$$D_{act} < D_{best} - T \cdot log(RND)$$

where the temperature T is a positive number and RND a random number between zero and 1. To get convergence, the threshold value `T` will be reduced after a pre-defined number of runs (see 1.5.13).

Variation rate and big variation rate

The variation rate r is the maximum variation for a positive parameter p during a run:

$$p = r \cdot rnd \cdot p$$

where rnd is a random number between 0 and 1. If negative values are allowed, the random number varies between -0.5 and 0,5. If the parameter value is 0, the parmeter is set to a random number between 0 and 1 resp -0.5 and 0.5.

If the exact optimization method is selected, the `Variation rate` increases to the `Big variation rate` if more than `Max. Steps` runs without improving the best parameter set are calculated. The variation is decreased after the next improvement.

Max. runs

The maximum number of runs without improvement until the optimization is reinitialized (EX). In this case the parameter values are reset to the values from the `.cev file`,

Max. steps

The maximum number of runs without improvement until the variation rate is increased (only EX).

The number of runs until the temperature resp. the threshold is decreased (TA,SA). If no improvement is reached for a specific temperature value, the system is reinitialized.In this case the parameter values are reset to the values from the `.cev file`. The temperature is set to its starting value.

Temperature, threshold and annealing rate

The temperature resp threshold T controls the optimization according to 1.5.13 resp.1.5.13 . The annealing rate r_a determines how fast the temperature resp. threshold decreases:

$$T = r_a \cdot T$$

[Start Rnd]

Starts the optimization. Beginning with the parameters shown in the `value` column the distance to the reference file is calculated. The next run is automatically started with a modified parameter set. The modifications amounts normally 10% of the absolute value of each parameter at maximum. If the value in the column `from` differs the value given in the column `to` it is additionally proved if the parameter lies in between this interval. If not the mean value of both boundaries is used as new parameter. If `Positive Par` is enabled only positive values are used. If a negative value is selected by random, the old value for the parameter is fixed for the next run.

If the new distance is smaller than the previous the parameters are set to this better values. This process is repeated until `[Stop Rnd]` is pressed. The actual best parameter set is stored in `bestpar.bat`, the distance and the difftype is stored in `bestdist.dat`, the run itself is stored in `best.outc`.

If the distance value could not be upgraded for ten runs modifications of 100% of the absolute value of each parameter at maximum are allowed until the distance is upgraded. If the distance could not be improved for more than 10000 runs the system will be reinitialized with the original parameter values.

1.5.14 [Stop Rnd]

Stops the optimization. The actual run will be finished.

1.5.15 [Reset to Best]

Sets the parameters and the distance to the last best values if exist. Reads the distance and the counter from file.

1.5.16 [Reset to AbsBest]

Sets the parameters and the distance to the absolute best values if exist. Reads the distance and the counter from file. The absolue best values and the best values differ if simulated annealing or threshold accepting is active.

1.5.17 [Rename]

All files belonging to a run can be renamed to keep it. The renamed files will be stored in the directory `<model_dir>/cevruns`.

1.5.18 [Reset]

Deletes all "'best"'-files (except the file `bestpar.bat`) in the directory `cevos`. After reset the files named `absbest*` are moved to the directory `cevos_absbest` in the model directory parallel to the directory`cevos` . The files get a time stamp. A reset should be done before changing the distance type (1.5.8).

1.5.19 [Tiger Graphics MoViE]

Starts the **TigerGraphics** tool **MoViE** and opens directly the compare tool for the reference file and the last best run. For further information see Kohlmeier & Hamberg (2004).

1.5.20 [Exit]

Closes the **CEvoS**-shell. The best-files are still available and can be read after restarting **CEvoS**.

1.5.21 [CEV File]

A `cev`-file determing the parameters can be defined. By default the file `cevos.cev` is read. After changing the file the parameter window will be new initialized.

1.5.22 [CIN File]

A `cin`-file controlling the simulation can be defined (see section 19.7 in Hamberg & Kohlmeier (2004) for details). If `[cancel]` is pressed, no `cin`-file is evaluated.

1.5.23 [Reference File]

A `.outc`-file can be defined. This is taken as the goal for the training of the model under regard. All distances (pointwise or L2) are computed against the values held in this file (see 3). If `[cancel]` is pressed, no reference file is selected and the calculation of distances will not work.

1.5.24 [Stop Running Model]

A running simulation can be stopped. If the random process is running **CEvoS** continues with the next parameter set.

1.6 The file cevos.init

The file <model_dir>/cevos/cevos.init is optional. If the file exist, the default settings are overruled. The file can contain the settings for the control of the optimization runs. It must be written in **Tcl/TK** notation. Example:

```
# cevos file with absolute path!!
# Default:  set cev_file $cev_resultdir/cevos.cev
 set cev_file $cev_resultdir/cevos.cev

# cin file with absolute path!!
# Default: none
# set cin_file

# Reference file with absolute path!!
# Default: none
# set cev_reference_file

#Optimization
#SA=0 exact EX
#SA=1 threshold acceptance TA
#SA=2 simulated annealing SA
set SA 0

# Variation rate of Parameters
set smallchange 0.1
#big variation of parameters (only SA=0)
set bigchange 1.0

#number of runs without change in best until variation
#is increased to bigchange (only EX)
#number of runs until temperature/threshold is decreased (only TA, SA)
set maxsteps 100

#  number of runs without change until system is reinitialized
set maxruns 1000
```

```
#distance type
#l2dis=0 point wise
#l2dist=1 L2-norm for Fourier series
set l2dist 0

#distances (cost function)
set difftype 1

# posval=0 all values
# posval=1 only positive parameters
set posval 1

#Initializing of des simulated annealing and threshold acceptance
#initial temperature (SA) resp. initial  threshold (TA)
set T 1

#decreasing rate for T (SA ,TA)
set rate 0.9
```

2 Output Files

2.1 best.outc, best.info, best.log

The file `best.outc` is updated whenever a better distance is reached. It contains the results of the last best run. Additionally the files `best.log` and `best.info` are written. For details see **CEMoS** manual (Hamberg & Kohlmeier, 2004).

2.2 best.dist

The file `best.dist` is created whenever a model run is finished. The first row contains the actual best distance, the distance type. The third row contains the number of runs. The file will be deleted by `[Reset]`. If no values for comparison are available the file contains dummy values (999999...).

2.3 best.bat and best.cin

The file `best.bat` is updated whenever a better distance is reached. It contains the best parameterization if form of a valid batch file. The parmeters are also given in cin-file format (`best.cin`, see Hamberg & Kohlmeier (2004)).

2.4 The files absbest*

The optimization with threshold acceptance (TA) and simulated annealing (SA) leads to best results which are not absolutely the best (this means that they have not the absolute minimum value for the distance but have been accepted according to the conditions of TA resp. SA). This is of special interest if several local minima of the cost function exist and the absolute best value and the best value are results from different local minima. Thus, for TA and SA the absolutely best results according to the conditions of EX are also stored in the files named `absbest*`. The fact, that both are not the same is a hint that the optimization algorithm does not have converged up to know. After reset the files named `absbest*` are moved to the directory `cevos_absbest` in the model directory parallel to the directory`cevos` . The files get a time stamp.

3 Converting field data to comparable files

Normally field data are not available for the whole simulation period and are mostly not equidistant. For the comparison with **CEvoS** is strictly necessary to definition a reference file which has the same structure as the result files itself. The easiest way to do this is to create the reference file by the model itself. In the case of field data this is normally not useful. The **TigerGraphics** tool **MoViE** Kohlmeier & Hamberg (2004) provides the possibility to convert field data to the format needed for **CEvoS**(see **MoViE** → Prepare Tool → [**data** → **outc**]). The values are linearly interpolated. The startime, endtime, outdelt and cycle must be set to the values of the model for which the reference is designed. If the interval of the resultfile exceeds the field data, the values for the missing time stamps at the beginning and at the end are set to -9999.0. All values which are exactly -9999.0 will not be considered during the comparison. If the comparison should be done with L2 distances the (linear interpolated) reference file is automatically converted to spectral values.

WARNING: Actually it is not possible to select a `_dpk.outc` file for reference!! (see **MoViE** → Prepare Tool → [**data** → **Fourier**]).

4 Auxillary scripts

4.1 def2cev

The script `def2cev` converts **CEMoS** definition files (extension .def) into **CEvoS** parmeter files (extension `.cev`). All lines not needed by **CEvoS** will be deleted. Thus, only the statments `%real_par` and `%real_ind_par` are evaluated. Lines defining `real_ind_par` are splitted into single lines, f.e.:

```
def2cev xyz.def
```

converts the file `xyz.def`:

```
%real_ind_par
ampli[3]={1,2,3 }; /*Amplitude*/
phase[3]={0,0.25,0.5 }; /*Phase*/
freq[3]={1,1,1 }; /*Frequency*/

%globals
sinus[3];

%change
phase[1-3]={0,0.5,0.5 };
```

to `xyz.cev`

```
ampli[1]=1;  /*Amplitude*/
ampli[2]=2;  /*Amplitude*/
ampli[3]=3;  /*Amplitude*/
phase[1]=0;  /*Phase*/
phase[2]=0.25;  /*Phase*/
phase[3]=0.5;  /*Phase*/
freq[1]=1;  /*Frequency*/
freq[2]=1;  /*Frequency*/
freq[3]=1;  /*Frequency*/
```

It is strictly recommended to manually correct the automatically converted files afterwards.

4.2 cev2ser

The script `cev2ser` converts a regular `cev`-file in parameter notation (see 1.3) to a `cev`-file in series notation. The script requires up to 3 arguments. First the filename to be converted, second the percentage for the modification of the parameter and third (optional) the flag `pos` if only positive (including zero) values are allowed: The file `xyz.cev` is converted by

```
cev2ser xyz.cev 10 pos
```

to the file `xyz_ser.cev`:

```
ampli[1]:0.9:1.1:0.1;/*Amplitude*/
ampli[2]:1.8:2.2:0.2;/*Amplitude*/
ampli[3]:2.7:3.3:0.3;/*Amplitude*/
phase[1]:0:0.1:0.1;/*Phase*/
phase[2]:0.225:0.275:0.025;/*Phase*/
phase[3]:0.45:0.55:0.05;/*Phase*/
freq[1]:0.9:1.1:0.1;/*Frequency*/
freq[2]:0.9:1.1:0.1;/*Frequency*/
freq[3]:0.9:1.1:0.1;/*Frequency*/
```

The first value of a row is the lower boundary, the second the upper boundary and the third the increment (only needed for series runs). If the original value is zero, the range is set to [-percentage,percentage]. If `pos` is set, negative values are changed to zero. Original negative values are set to zero if `pos` is set.

References

Hamberg, F., & Kohlmeier, C. 2004. *CEMoS User's manual*. `http://www.staff.uni-oldenburg.de/cora.kohlmeier/software/cemoscemtk.pdf`.

Kohlmeier, C, & Hamberg, F. 2004. *MoViE User's manual*. `http://www.staff.uni-oldenburg.de/cora.kohlmeier/software/movie.pdf`.

Krug, P. 1992. *Abstandsfunktionen in Hilberträumen und Schätzfunktionen in separablen Banachräumen mit Anwendungen in der mathematischen Modellierung*. Aachen: Verlag Shaker.

Morgan, B.J.T. 2001. *Applied Stochastic Modelling*. London: Edward Arnold.

Spitz, Y. H., Moisan, J. R., Abbott, M. R., & Richman, J. G. 1998. Data assimilation and a pelagic ecosystem model: parameterization using time series observations. *Journal Of Marine Systems*, **16**(1-2), 51–68.

Stearns, S.D. 1984. *Digitale Verarbeitung analoger Signale*. 2 edn. München Wien: Oldenbourg.